Interacting with robots via speech and gestures, an integrated architecture

Francesco Cutugno, Alberto Finzi, Michelangelo Fiore, Enrico Leone, Silvia Rossi

Dept. of Electrical Engineering and Information Technology
University of Naples "Federico II" - Italy
{francesco.cutugno, silvia.rossi, alberto.finzi}@unina.it

10.21437/Interspeech.2013-587

Abstract

Effective human-robot communication is one of the main concerns in modern robotics. Involved systems should be very robust, allowing little chance for misunderstanding users commands. The main purpose of this work is to develop a general framework for multimodal human-robot communication, which allows users to interact with robots using speech and gestures, integrated into unique commands. The produced architecture relies on the definition of different modules separately analysing the low level inputs and presenting a further fusion module able to extract semantics from these multiple channels. In this paper, we introduce our general approach and provide a case study where gesture and speech modalities are combined.

Index Terms: speech commands, gesture, multimodal interaction

1. Introduction

In order to work with humans, a robotic system should be able to understand users commands and intentions interacting within a given workspace [1]. In a multimodal interaction context, users can communicate with a robot using several input channels, called modes, which are analysed and integrated by the system. The redundancy introduced by the different modes can solve faults of one or more channels, hence, the robustness is generally enhanced. This redundancy can be useful for different purposes. In noisy environments, the information provided on a channel could be unreliable and redundancy can help reducing errors in the interpretation of the input signals, e.g. in a low light environment speech commands may be more reliable than gestures [2]. Moreover, in a system with different interaction contexts, the various modalities could be more or less relevant depending on the task. In these cases, weighting inputs in different ways could produce a more efficient communication. Finally, multimodal interaction should be more intuitive and natural, since humans normally interact with the environment and with other subjects in a multimodal way.

In this paper, we propose a general architecture supporting multimodal human-robot communication and interaction. The proposed architecture combines different layers of classification and interpretation, combined together with the goal of being able to give simple commands to a robot. We introduce our general approach and provide a case study where we consider gesture and speech as the main input channels, as it is nowadays expected in Human-Robot Interaction (HRI).

The first multimodal system proposed in the field of Human-Computer Interaction (HCI) is Bolt's "Put-That-There" system [3], that allowed a user to interact with shapes on a monitor using speech and gesture commands. Starting from Bolt's approach, researchers have continued to develop multimodal systems, trying to achieve simpler and more efficient ways to interact with computers (among the others, see [4, 5]).

In the field of HRI, many approaches exploit multimodal interaction to improve human-robot communication. However, in many of these systems, speech is usually the dominant modality, with gestures being mainly used for the disambiguation of objects. For example, in [6], an HRI architecture for multimodal fusion in a kitchen scenario using speech and gesture inputs was presented. This work considered speech as the main input modality while pointing gestures were used mainly for object disambiguation. In [7], a system designed to handle natural artifacts performing 3D gestures using a stereo camera system and a bank of collaborative particle filters was presented. This approach considered speech as the main input modality too. When the speech interpreter needed a complementary gesture for disambiguation, the system applied a rule-based fusion at decision level with the gesture interpretation results provided on the same time window. Differently from these approaches, we do not assume dominant input modalities as a user should be able to interact by using both multimodal and monomodal commands. Fusion engines can be designed both as rule-based [6, 7], although this approach is uncommon in multimodal HRI systems, and as stochastic classification systems. On the other hand, these latter systems, and in particular Support Vector Machines (SVMs), were proven to be very efficient and to produce better results than rule-based systems or other machine learning algorithms [8]. Our approach tries to integrate a multimodal fusion on an HRI problem by using SVMs.

The paper is organized as follows: in Section 2 we introduce our system, in Section 3 we illustrate the test-bed, the various training corpora and the training techniques; in Section 4 we will give some sketches of tests results; finally, in Section 5, we discuss conclusions and possible future developments.

2. The proposed architecture

The proposed system is showed in Figure 1.

Separate processes control the respective modalities and classify the features extracted from the raw data provided by the different input sensors. The system uses a multimodal late fusion strategy and is divided in different layers. Both recognition axes classify features extracted and create a N-best list of possible interpretations. These are then synchronized and sent to a Fusion Engine constituted by an SVM classifier [9].

2.1. Gesture recognition

The visual channel, dedicated to gesture recognition, is based on the Microsoft Kinect™technology. Kinect is able to recognize and track a number of different users in a scene, separat-

Authors appear in strict alphabetic order.

Copyright © 2013 ISCA

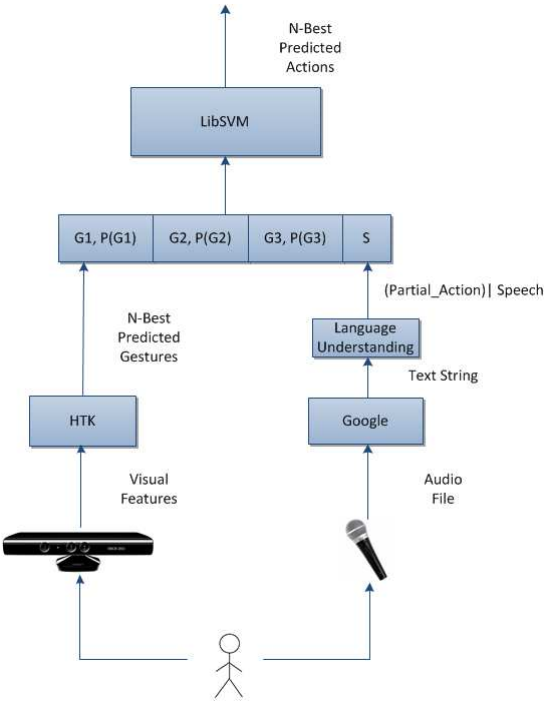

Figure 1: architecture of the multimodal system

ing them from the background items. The system searches for shapes corresponding to a generic humanoid template on which two arms, two legs, a head, a torso can be found in the correct proportions. Kinect represents users with a skeleton composed of 20 joints and it is able to track user movements in real time, discriminating among 200 different skeleton poses. The output of the Kinect sensor is captured by an interface module designed to recognize human gestures. This module is implemented by using the OpenNI libraries (http://www.openni.org/). As the data stream coming from the Kinect is noisy, the performance of the gestures classifier is low when raw data are used as features. This problem is addressed in our system by applying the Kalman filter implementation available in the OpenCV (Open Source Computer Vision, http://opencv.org/) Library. By using the Kalman filter, a better estimate of the joints movements can be obtained from the raw positions tracked by the Kinect sensor. We used the Kalman Filter to estimate the 9 parameters corresponding to the 3D points of the user's wrist, elbow and shoulder. For gesture recognition, Kinect directly provides features that are able to represent a gesture trajectory and tolerant to shifts in location, to scale and to rotations.

Our feature-set includes:

- The 3D coordinates of the shoulder, elbow, and hand joints, calculated by Kinect.
- The 3D angles between the shoulder and the elbow.
- The 3D angles between the elbow and wrist.

leading to a partial set of 15 features. We specify our points as angles θ in the polar coordinates space. The coordinates are normalized so that $0 < \theta < 8$ to reduce noise, as shown in [10]. The representation given by the features accounts for different execution speeds for the expected gestures.

In the virtual skeleton provided by the Kinect, hands are not tracked. As the fingers movements and palm orientation are very important for hand gestures, we decided to use a colored glove, see Figure 2. In this glove each finger, the palm and the back of the hand are painted with a different color.

Finger shapes are then detected by means of color threshold detection algorithms. To minimize false positives, we isolated the user's right hand in the image by reading the positions of the joints corresponding to wrist and by creating a squared sub-image centered on the predicted position. The length of this square side was computed equal to half of the distance between the wrist and the elbow. Once again we recurred to OpenCV libraries to detect the different colors on the glove, to convert the RGB image to the HSV (Hue Saturation Value) color space, using the OpenCV function *cvtColor*. In our task we need to distinguish among three different hand shapes, namely "open", "closed" and "pointing". These positions are obtained by recognizing different color blob contours for the index finger (pointing), the hand palm (open position) and back or four colored blobs grouped together in shape of fist (closed position). For the hand position we add two further features to the above listed set, in particular:

- An integer value representing the hand type (open, closed, pointing).
- A boolean value, that indicates if the hand is directed with the palm or back toward the camera.

The final set for gesture recognition is then composed of 17 features acquired 18 times per second.

Two different classification systems have been tested for gesture recognition.

2.1.1. HMM gesture recognition

The first system is based on Hidden Markov Models as implemented in HTK [11]. Even if HTK is primarily used for speech recognition tasks, it is well known that it can be adapted to other applications, like character recognition [12], DNA sequencing [13] and gesture recognition [14]. To use HTK for gesture recognition, we created a set of HMM using a double-Bakis model with three hidden states, plus non-emitting starting and ending state and three Gaussian Mixtures on each state.

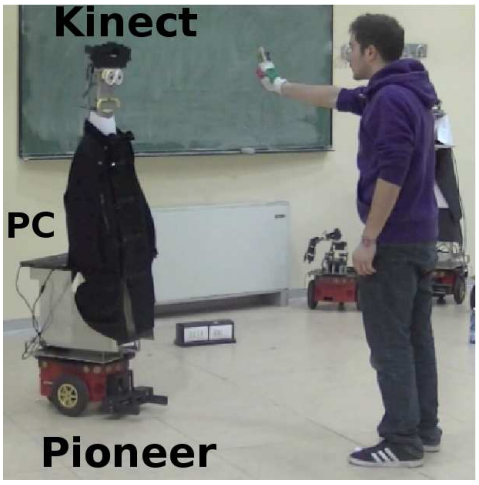

Figure 2: tester using a glove for hand pose recognition

2.1.2. LDCRF gesture recognition

The alternative gesture classification system is based on LDCRF (Latent Dynamic Conditional Random Fields) [15]. The chosen protocol fixed at 7 the number w of considered analysis frame and evaluated the use of (h =5, 6 and 7) hidden states. We used the HCRF Library (http://sourceforge.net/projects/hcrf/). As we will see in the results section, LDCRF gave good performances and they deliver an higher number of possible outputs, as they make a gesture guess at each frame of the pattern tracking. This could be used in the future to take decision in particular cases of ambiguity.

2.2. ASR

For the speech-to-text conversion, we use Google speech tools, introduced in the Chrome browser from version 11 onward. Google speech tools is a web service which receives in input an audio file in FLAC format and returns a JSON (JavaScript Object Notation) object containing the text string decoded from the audio file and its confidence level. We choose to use this off-the-shelf product, rather than an HTK based system developed internally, as we aimed at having a speaker and vocabulary independent system, in order to have, in the future, no constraints concerning the naturalness of speech interaction. The computational load required to extract the given semantics from this modality was consequently transferred to the SLU module (see Section 2.3). To handle audio Input/Output, we use PortAudio [16], a free and cross-platform library. We submit the collected audio files to Google servers by means of an HTTP POST and analyze the object (1-best sentence with its confidence probability) returned by the web service with a SLU module.

2.3. Spoken Language Understanding

The SLU unit analyzes the ASR output to extract meaningful information, recurring to the framenet technique [17, p.41]. This analysis process is hierarchical and based on the use of different SVM units. In order to extract the features needed for training and classification, each string unit is pre-processed using Tree Tagger (http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/), in order to to extract part-of-speech and lemma information from each sentence. We keep only those words that represents nouns, verbs, or adjectives. Words are codified with indexed integers. At the top level of our SLU hierarchy, a SVM [18], called Frame SVM, implemented recurring to the LibSVM [9] libraries, analyzes the preprocessed text string to choose the correct frame to represent the information. We define two different kinds of frames:

- SimpleAction: which represents commands such as "Yes" or "No" that do not need additional information. Its slot is:
  - Name: that accepts a string value.
- DirectedAction: represents commands like "Take" or "Drop" that may need additional information. Its slots are:
  - Name: that accepts a string value.
  - Object name: a string value representing the name of the target object.
  - Object info: a string value that represents additional information, like the color of an object.
  - Object location: a string value representing the position of the object.

Point at	Take
Drop	Don't
Give	Come here
Search	Stop

Table 1: List of gestures

Follow me
Take that [$COLOR_LIST][$OBJECT_CLASS]
Drop the [$OBJECT_CLASS]
Don't
Give me the [$OBJECT_CLASS]
Come here
Search that area
Go there
Stop

Table 2: Possible robot actions

The Frame SVM is trained on a vector of features representing the coded words after preprocessing and filtering. The Frame SVM returns a single word (or its class of inclusion) if it detects a *SimpleAction*, alternatively, it invokes a further SVM unit, called SVM Tagger, if a *DirectAction* is returned. SVM Tagger fills the semantic slot tags in the *DirectAction* frame. This unit is trained with a list of single words (presently representing istances of Object_Class and Color_List (see Table 2). The SLU module returns a 1-best guess on the possible partial verbal action intended by the user.

2.4. Fusion

The Fusion Engine consists of another SVM classifier receiving in input a quadruple formed by the 3-best guess from the gesture recognition channel, and a guess on a possible verbal action (represented by the corresponding frame) as delivered by the SLU module. With this quadruple as input, the SVM returns the complete N-best list of hypotheses for executable robot actions.

3. Testing scenario - Training corpora

Looking at our scenario, we have defined the set of gestures listed in Table 1.

Integrating speech utterances with the above listed set of gestures leads to the definition of the pseudogrammar shown in Table 2 which represents the entire set of possible robot actions corresponding to the human command:

$COLOR_LIST and $OBJECT_CLASS respectively represent non-terminal symbols indicating terms that user can freely utter from a virtually non finite list of real objects (i.e. ball, bottle...) and colors.

3.1. Gesture Corpus

For each gesture, we collected 16 samples from 10 different users, five males and five females, obtaining a training set of 1280 elements. Before each recording session, users were trained by showing them an expert performing each gesture, so that they could perform them in a similar way.

3.2. SLU Training

Since the ASR module is based on a Google tool, we only trained hierarchical SVM system for SLU. For this purpose,

	Point at	Come here	Give	Search	Take	Drop	No	Stop	Go	P	R	F
Point at	7,8	0,1	0,1	0	0	0	0	0	0	1	0,97	0,98
Come here	0	10,5	0,3	0	0,2	0	0	0	0	0,98	0,95	0,96
Give	0	0	12,8	0	0,2	0	0	0	0	0,87	0,98	0,92
Search	0	0	0,1	8,8	0,1	0	0	0	0	1	0,97	0,98
Take	0	0,1	0,8	0	22,1	0	0	0	0	0,96	0,96	0,96
Drop	0	0	0,5	0	0	10,5	0	0	0	1	0,95	0,97
No	0	0	0	0	0,2	0	7,6	0,2	0	0,98	0,95	0,96
Stop	0	0	0	0	0	0	0,1	12,9	0	0,98	0,99	0,98
Go there	0	0	0	0	0	0	0	0	4	1	1	1
Average										0,97	0,97	0,97

Table 4: Multimodal Testing, average accuracy 0.97%

we used a corpus of 200 written sentences coherently designed to be part of the complex command made of a combination of speech and gesture. The amount of possible terms and generable sentences chosen for the test scenario is to be considered as preliminary and quite limited. We anticipate in this section results for evaluation of this branch of our model as it is, given these facilitated conditions, quite close to 100% of success, in the sense that SLU module was able to compensate for the really rare errors provided by the Google ASR.

3.3. Fusion Engine Training

To train the Fusion Engine we produced a set of 1000 coherent feature vectors (the quadruple described in Section 2.4) automatically generated. For each quadruple, the correct class identifying the robot action, was manually added.

4. Results

In this section we give an overview of the results obtained in each module both separately and combined.

4.1. Gesture Recognition

We present here results of the two alternative implementations of statistical gesture recognition systems.

4.1.1. HMM

To evaluate the HMM used for gesture classification we performed the 10-fold cross validation on our data set of 1280 elements. The model achieves an average accuracy of 0.65% on the whole set of gestures. We produced a confusion matrix (not reported here) where it could be seen that some gestures, like Search vs. Don't or Take vs. Drop, are often misclassified. This was expected because we defined the gestures to be ambiguous, to increase the naturalness of the system and use the capacity of the dialogue manager to disambiguate them.

4.1.2. LDCRF

In Table 3 we report results, obtained with the same test procedure used for HMM, given by the LDCRF gesture recognition system. We varied both w and h parameters.

As expected, LDCRF behave better than HMM, as it can be seen the highest accuracy (72.94) is obtained for $h = 7$ and $w = 7$. The analysis of the confusion matrix shows almost the same gesture misinterpretation, but with lower confusion percentages. We decided to use LDCRF and, in the following, we assume this approach.

	$h = 5$	$h = 6$	$h = 7$	$h = 8$
$w = 5$	66,82	67,44	68,95	67,90
$w = 6$	69,44	69,90	69,10	71,14
$w = 7$	70,54	71,55	72,94	72,92

Table 3: Accuracy results for LDCRF Gesture Recognizer

4.2. Multimodal Classification

As reported in Section 2.3, for this relatively simple task, made of a limited number of actions to recognize, performances of the SLU module were close to 100%. Next, we evaluated the SVM model used for the multimodal fusion. To do this, we performed 10-fold cross validation on a set of further 1000 elements (as previously described in Section 3.3). The model generalizes well to independent data, achieving an average accuracy of 0.97. Results are shown in Table 4. Terms corresponding to non-terminal symbols in Table 2 are not indicated.

5. Discussion and conclusions

We presented a multimodal framework for natural, robust, and flexible human-robot communication and interaction. Most of the multimodal HRI systems proposed in literature rely on a dominant modality while, in contrast, our system allows the users to express their instructions as combinations of gestures and speech inputs. For this purpose, we introduced a multilayered process based on a late fusion approach. In this context, speech and gesture commands were interpreted exploiting a fusion approach based on SVM. To provide a more natural experience the coloured glove should be removed and the finger detection algorithm modified to work without color blobs detection. Extending the set of commands that can be given to the robot will provide further data on the robustness of the speech branch of the system, as the dataset used for the presented experiments is still limited. Further investigations are in course to manage a wider range of communication modalities. In order to stress system performances we will design a test corpus affected by some sort on noise aiming at better describing interactions between different modalities and how they cooperatively recover the user intended meaning.

6. Acknowledgements

This work has been supported by EU FP7 "SAPHARI" under grant agreement no. ICT-287513.

7. References

[1] A. D. Santis, B. Siciliano, A. Luca, and A. Bicchi, "An atlas of physical human-robot interaction," *Mechanism and Machine Theory*, vol. 43, no. 3, pp. 253–270, 2007.

[2] A. Bannat, J. Gast, T. Rehrl, W. Rösel, G. Rigoll, and F. Wallhoff, "A multimodal human-robot-interaction scenario: Working together with an industrial robot," in *Proc. of the 13th Intern Conf on HCI. Part II: Novel Interaction Methods and Techniques*. Springer-Verlag, 2009, pp. 303–311.

[3] R. A. Bolt, ""put-that-there": Voice and gesture at the graphics interface," *SIGGRAPH Comput. Graph.*, vol. 14, no. 3, pp. 262–270, Jul. 1980.

[4] P. K. Atrey, M. S. Kankanhalli, and J. B. Oommen, "Goal-oriented optimal subset selection of correlated multimedia streams," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 3, no. 1, Feb. 2007. [Online]. Available: http://doi.acm.org/10.1145/1198302.1198304

[5] B. Dumas, D. Lalanne, and S. L. Oviatt, "Multimodal interfaces: A survey of principles, models and frameworks," in *Human Machine Interaction*, ser. Lecture Notes in Computer Science, D. Lalanne and J. Kohlas, Eds. Springer, 2009, vol. 5440, pp. 3–26.

[6] H. Holzapfel, K. Nickel, and R. Stiefelhagen, "Implementation and evaluation of a constraint-based multimodal fusion system for speech and 3d pointing gestures," in *Proc of the 6th Intern Conf on Multimodal Interfaces*, ser. ICMI '04. ACM, 2004, pp. 175–182.

[7] B. Burger, I. Ferrané, F. Lerasle, and G. Infantes, "Two-handed gesture recognition and fusion with speech to command a robot," *Auton. Robots*, vol. 32, no. 2, pp. 129–147, 2012.

[8] J. Fiérrez-Aguilar, J. Ortega-Garcia, D. Garcia-Romero, and J. Gonzalez-Rodriguez, "A comparative evaluation of fusion strategies for multimodal biometric verification," in *AVBPA*, ser. Lecture Notes in Computer Science, J. Kittler and M. S. Nixon, Eds., vol. 2688. Springer, 2003, pp. 830–837.

[9] C. Chang and C. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on intelligent systems and technology*, vol. 2, pp. 27:1 – 27:27, 2011.

[10] X. Wenkai and E.-J. Lee, "Continuous gesture trajectory recognition system based on computer vision," *Applied Mathematics & Information Sciences*, vol. 6, pp. 339–346, 2012.

[11] S. Young, G. Evermann, and et al., *The HTK Book*, 2006.

[12] M. Khorsheed, "Offline recognition of omnifont arabic text using the hmm toolkit (htk)," *Pattern Recognition Letters*, vol. 28, no. 12, pp. 1563–1571, 2007.

[13] W. N. Grundy, "Modelling biological sequences using htk," Technical Report, prepared for Entropic Research Laboratory, Inc, Tech. Rep., 1997.

[14] T. Starner, J. Weaver, and A. Pentland, "Real-time american sign language recognition using desk and wearable computer based video," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 12, pp. 1371–1375, 1998.

[15] L. Morency, A. Quattoni, and T. Darrell, "Latent-dynamic discriminative models for continuous gesture recognition," in *Proc. of CVPR*, 2007, pp. 1–8.

[16] R. Bencina and Others, "PortAudio – portable cross-platform Audio API," 2008. [Online]. Available: http://www.portaudio.com

[17] G. Tur and R. De Mori, *Spoken language understanding: Systems for extracting semantic information from speech*. Wiley, 2011.

[18] T. Hastie, J. Friedman, and R. Tibshirani, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2006.